```
I ← 0;
repita
|  A ← B + C;
|  B ← C;
|  I ← I + 1;
até I > 18;
```

teríamos:

```
I: = 0;
repeat
   A: = B + C;
   B: = C;
   I : = I + 1;
until I > 18
```

ou, se quisermos utilizar o comando enquanto:

```
A: = B + C;
B: = C;
I : = 1;
while I <= 18 do
   begin
      A: = B + C;
      B: = C;
      I : = I + 1;
   end
```

f) REPETIÇÃO (*para-faça*)

PORTUGOL

```
para <var> de <exp1> até <exp2>
|                passo <exp3> faça
|      <lista de comandos>
fim para;
```

PASCAL

quando o passo for igual a 1

```
for <var> : = <exp1> to <exp2 K do <comando>
```

quando o passo for igual a − 1

```
for <var> : = <exp1> downto <exp2> do <comando>
```

Neste comando, var, exp_1, exp_2 *não podem ser modificados* dentro do laço e os passos só podem ser 1 e − 1. Quando os incrementos forem diferentes de 1 ou − 1, recomenda-se a utilização do comando *while* em vez do comando *for*.

Vamos passar para o PASCAL:


```
para I de 1 até 20 passo 1 faça
|  para J de 20 − I até 1 passo − 1 faça
|  |  A ← A + 1;
|  |  B ← B + J;
|  fim para;
fim para;
```

Solução:

```
for I: = 1 to 20 do
   for J: = 20 − I downto 1 do
      begin
         A: = A + 1;
         B: = B + J;
      end;
```

Problema:

Passar para o PASCAL:

```
para I de 1 até 10 passo 2 faça
|  M ← M + 2;
|  J ← I + 2;
fim para;
```

Solução:

```
I: = 1;
while I <= 10 do
   begin
      M: = M + 2;
      J : = I + 2;
      I : = I + 2;
   end;
```

i) ALTERNATIVA MÚLTIPLA ESCOLHA (*escolha*)

PORTUGOL

```
escolha <expr>
|  caso <c11>: <c12>: . . . : <c1n>:
|               <lista de comandos 1>
|  caso <c21>: <c22>: . . . : <c2n>:
|               <lista de comandos 2>
|   .
|   .
|   .
|  caso <cn1>: <cn2>: . . . : <cnn>:
|               <lista de comandos n>
|  senão <lista de comandos n + 1>
fim escolha;
```